

МИНОБРНАУКИ РОССИИ  
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ  
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)  
Кафедра информационных систем

КОНТРОЛЬНАЯ РАБОТА  
по дисциплине «Объектно-ориентированное программирование»  
ТЕМА: Интерфейс курсовой работы.

Студенты гр. 4372

Климаш И.В.

Преподаватель

Егоров С.С.

Санкт-Петербург

2025

Студент Климах И.В.

Группа 4372

Исходные данные: Разработать графический интерфейс для курсовой работы номер 13.

Компьютер с  $N$  (3) процессорами должен решить  $M$  (16) заданий. Каждое  $i$ -ое ( $i=1..M$ ) задание может выполняться на любом процессоре без прерывания своей работы за некоторое априори заданное время. Это время генерируется в начальном состоянии случайным образом на интервале  $T_1 — T_2$  (2-8) с равномерным распределением вероятностей. Предварительно все задания пользователь выстраивает в очередь. После этого задачи запускаются на выполнение, т.е. Делается прогон. Первая в очереди задача становится на выполнение на первом свободном процессоре без какой-либо задержки. Работа заканчивается при выполнении последнего задания, при этом для прогона фиксируется распределение задач в исходной очереди и суммарное время их выполнения. Состояние системы при самом прогоне должно отображаться в реальном времени. Прогон можно начать заново, формируя другую очередь при сохранении времен выполнения заданий. Если время выполнения при очередном прогоне будет лучше, то его параметры (последовательность и суммарное время) заменяют лучший предыдущий результат.

Следует также предусмотреть переход в начальное состояние (без перезапуска приложения), в котором задаются новые параметры системы (не обязательно все).

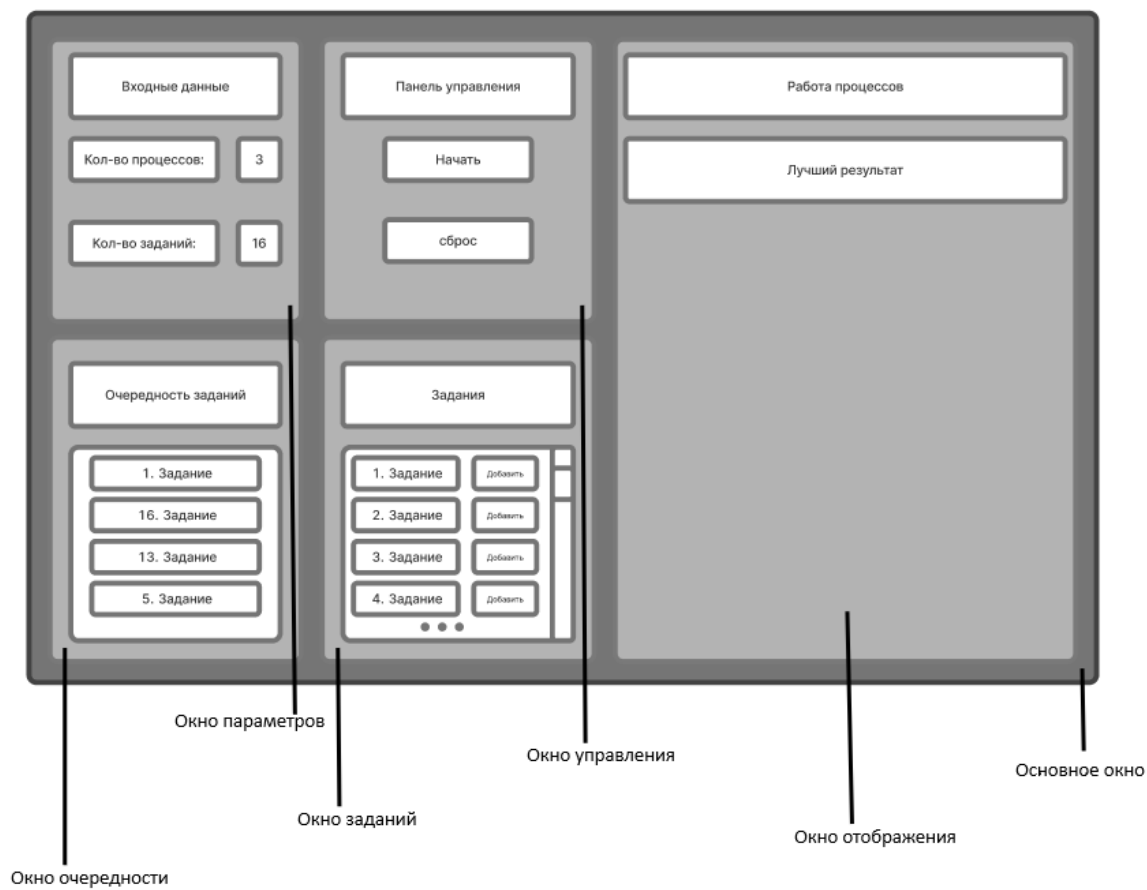


Рисунок 1 – Графический интерфейс.

## **ВЫВОДЫ ПО ВЫПОЛНЕНИЮ РАБОТЫ**

В рамках выполненной работы был успешно разработан и графически представлен в виде прототип интерфейса, включающие требуемые окна в конфигурации SDI, где окна параметров, управления и отображения композиционно связаны с главным окном, что соответствует заданным классам, без совмещения их функционала на данном этапе, что обеспечивает четкое разделение ответственности и служит готовой основой для последующей программной реализации.